ROBOT PATH PLANNING USING A GENETIC ALGORITHM

Timothy F. Cleghorn
Paul T. Baffes
Lui Wang

Technology Development and Applications Branch/FM7
Mission Planning and Analysis Division
NASA/Johnson Space Center
Houston, Texas 77058

Abstract:

Robot path planning can refer either to a mobile vehicle such as a Mars Rover, or to an end effector on an arm moving through a cluttered workspace. In both instances there may exist many solutions, some of which are better than others, either in terms of distance traversed, energy expended, or joint angle or reach capabilities. A path planning program has been developed based upon genetic algorithm. This program assumes global knowledge of the terrain or workspace, and provides a family of "good" paths between the initial and final points.

Initially, a set of valid random paths are constructed. Successive generation of valid paths are obtained using on e of several possible reproduction strategies, similar to those found in biological communities. A fitness function is defined to describe the "goodness" of the path; in this case including length, slope, and obstacle avoidance considerations.

It was found that with some reproduction strategies, the average value of the fitness function improved for successive generations, and that by saving the best paths of each generation, one could quite rapidly obtain a collection of "good" candidate solutions.

Introduction:

Robotics operations in the 1990's and beyond will be characterized by increasing levels of system autonomy. This implies that the various algorithms being developed now under the guise of Artificial Intelligence will be applied to those robotics operations currently performed or controlled by human operators. It has became traditional to classify robotics operations according to such labels as: manipulation, path planning, sensors, end effectors, etc.; and to deal with these sub-areas independently. However, they are clearly linked, and in some cases solution approaches in one sub-area can be applied to problems in another. A characteristic found in many areas is that there exists a plethora of solutions, and that the problem then reduces to how to distinguish the good solutions from the not-so-good ones.

The robot path planning problem is such a case; and in fact the path can refer either to that of a mobile vehicle such as a Mars Rover or arm platform on the space station (MRMS), or to an end effector on an arm moving through a cluttered workspace. There often exist a large, (even infinite), number of paths between the initial and final positions, and the desire is not necessarily to determine the best solution, but to obtain a "good" one within some reasonable time frame. In the present discussion, we will assume a "global" approach; that is, the space through which the robot moves is fully modeled. The obstructions are assumed to be known, and initially are assumed to be static. The case of dynamic obstacles will be discussed in a later paper.

There are a number of approaches to this problem, most of which attempt to identify the absolute best path. The problems with these are that the computational times become excessively long, and that "best" is often a very subjective quality. What we have attempted to do is to locate a class of "good" solutions, from which one can be selected arbitrarily. The method we used was what has become known as the "Genetic Algorithm", [1].

The Genetic Algorithm (GA) is based upon fairly simple biological/evolutionary principles. A random initial population is generated, and allowed to evolve in such a way that the desirable characteristics of individuals are on the average enhanced, and the undesirable characteristics suppressed. The characteristics of any individual are described as a bit string, or concatenated bit strings, much as chromosomal material is composed of series of genes. Associated with each string is some "fitness" value which manifests itself as part of the overall fitness of the individual within that particular population. If the strings with high fitness values are retained, and combined with other strings of high fitness, while those of low fitness are allowed to disappear, the average overall fitness of the population should increase.

The mechanism by which the high fitness information is transferred is called "genetic crossover". As in biology, when two individuals mate, the offspring carry some genetic material from each parent. This is called crossover, and can be illustrated as follows:

Parent 1 : 101101¦0111	child 1 : 1011010010
X ¦	and/or
Parent 2 : 100011¦0010	child 2 : 1000110111

Thus a single crossover can produce two new individuals with genetic characteristics similar to their parents. Multiple crossovers obviously can produce many distinct children, but for the present application this was not found to be necessary. What did turn out to be necessary was the inclusion of an environmental driver, specifically, mutation. In simplest terms, this means that any bit has a low but non-infinitesimal chance of being flipped; a 1 becomes 0 and vice-versus. In our application, a mutation had prefound and often catastrophic effects upon the survival of the individual, similar to what is found in many natural mutations.

It is necessary to emphasize that the biological analog has limitations, and in one very important respect, we rewrote high-school biology. Most previous work on genetic Algorithms have used bit-strings which were of constant length. Ours not

only varied, but the variation itself was used as the fitness function, and hence as the driver for our "natural selection" law. A second major departure from previous work was the use of "Deification", the process by which we saved the best individual or individuals of each generation through subsequent generations; exempting them from mutation but allowing them to participate in the genetic crossover process. We did, as shall be explained in detail, retain a probabilistic approach: even the individual with the best fitness value can suffer an accident and hence fail to reproduce; although most of the time, the "good" genetic material was maintained and developed.

The Algorithm:

The following terms and definition will be used to describe the genetic algorithm.

Population: The set of valid paths (solutions)

Generation: The subset of the population which is under consideration at a given point in time.

Individual: A single member of the population; in this case, a single valid path. The individual is represented as a bit string, or possibly an ordered list, eg: 1011001011.

Fitness: Those properties by which the value of the individual is measured. Examples: total path length compared to other individuals, avoidance of steep inclines, energy expenditure, etc.

In order to perform the path-planning task for a mobile robot, the following general approach was taken. First, the terrain map was described. Then an initial group of valid paths were formed, using a random path generator. These paths were treated as the initial generation, and one of several strategies for reproduction was chosen. Successive generations were formed using the reproduction strategy; the process terminating after some arbitrary number of generations, usually 50. This either allowed for convergence to a set of "good" paths, or provided an indication that the strategy chosen was not convergent, i.e., the average and best fitness values failed to improve with time. During the run, the best path solutions in each generation were stored, allowing the selection of the best individual for the total population, even if it did not happen to appear in the final generation. This selection was made simply by choosing that individual path with the best fitness value.

Figure 1 illustrates one of the terrain maps. The black squares represent regions of exclusion, such as boulders. Paths were constructed as moves between adjacent white squares, either laterally or diagonally. It was permissible for the robot to pass between the corners of diagonally adjacent obstacles, but not between their laterally adjacent sides. The number within each square represents the "elevation"; so hills, valleys, craters, and canyons could be represented as well. Cliffs were represented as obstacles, because it was assumed that they were unclimbable, and that falling off of one would terminate the path. The numbers of rows and columns could be adjusted, so arbitrarily accurate maps could be drawn, depending upon the patience of the user.

It should be mentioned, however, that autonomous vehicles generally do not have true global information, but can "see" obstacles only within their immediate vicinity. Thus "global" means limited to the field of view. This does not invalidate the method for larger areas; it simply requires that intermediate goals be established, and that the algorithm be rerun until the final goal is attained. Except for pathological cases, back tracking should not be required to extend past the preceding sub-goal's origin. Consequently, a grid of about 15X15 was chosen as a compromise between computer time, user input time, and the desire for fineness of detail.

The initial set of paths for each run were constructed by a random path generator. Each initial path had to be valid; that is, it had to start at the user-selected origin, and terminate at the goal, and could not traverse obstacles. Otherwise, loops, hill climbs, crater traversals, etc., were allowed. The number of initial paths was set by the user.

Each valid path, or individual, had a fitness value, which determined that individual's status within its generation, which in turn determined how likely that individual was to reproduce, thus passing the "good" information on to the next generation. The fitness value was determined by: 1) length - the longer the path, the poorer the value; and 2) "energy" expended. The latter was modeled with a hiker in mind; going straight up a steep slope is much more difficult than is going up a switchback. To encourage the vehicle to take a gentle but longer slope up a hill, as opposed to a short but steep climb, a penality function was devised to be the square of the slope between adjacent points. Figure 2 illustrates two paths to the same goal, and shows their respective fitness values.

Traversals of descending slopes, (not cliffs), required no special energy outlay, so they were not penalized in the fitness values.

Therefore, a number was assigned to each of the initial paths, based upon that path's length and upon its energy efficiency. It should be reemphasized that only valid paths were permitted. The next step was to form a new generation, the average fitness value of which was hopefully smaller (i.e., better) than the original generation's average fitness value. To do this, required selection of one of several possible strategies. Attempts were made to pattern these after zoological analogues, with overlays from various societies, real or mythological. The choice of strategy determined the number of individuals permitted to mate, the number to be culled, the number to be "deified", - as well as the environmental pressure, the rate of mutation.

Mating pair selections and cullings were performed by using a weighted random selection. A "line" was formed, composed of the concatenated fitness values of the individuals within the generation. The line segments were normalized to unity, the value for th individual with the best fitness within that generation. Thus a worse path has a lower numerical value. The "line" is the summation of these segments. If it were intended that 10% of the generation should be culled, a number of "darts" equal to 90% of the generation size would be thrown at the "line". The longest segments have the greatest chance of being hit by a dart; that is, they are selected for survival. Of course, it is possible to miss the "best" in the generation - this happens in nature - the lead elk slips and fractures a leg, or runs afoul of a predator. On the average, however, the least fit of the generation will be eliminated. Subsequently, mating pairs are formed. A "survivors line" is formed in a similar manner, and the "darts" thrown again. Mating pairs are established by pairs of dart throws. As a result, on the average, the best fit will be paired together, although again there is a random element.

Two additional points should be made here. First, it was required that two individuals be selected. If the line segment

representing a single individual was chosen twice within the same mating pair, another "dart' was thrown, until a second individual was obtained. Second, and of great importance for the correct functioning of the G.A., an individual could be selected for more than one mating pair. This is the scheme by which the "good" individuals produce more offspring on the average.

The mating of two individuals leads to genetic crossovers in their offspring. In each case, both possibilities formed from a single crossover were enrolled into the next generation. In practice, when two individuals were selected, a search was made for a common grid point. If several points were found in each individual, one would be selected at random, and the initial portion of each path would be concatenated with the trailing portion of the other. If no common points were found, the individuals would be enrolled as is in the new generation. Thus, in either case, two individuals were added to the next generation. The number of pairs to mate determines the number of offsprings, so the overall size of each generation will shrink, expand, or stay the same according to the strategy chosen by the user. The usual practice was to keep the size of the generation constant, in order to prevent either premature extinction prior to the evolution of "good" paths, or the overloading of the computer by too many individuals. This was done by "filling" the subsequent generation with members of the mating pool, based again on a random weighted selection process.

An additional ingredient was found to be necessary in order to prevent what resembled a collapse of genetic diversity. If a small, but finite mutation rate was not added, after several generations a small group of identical paths was formed. Our "mutation" kept the pot boiling, so that new solutions could be obtained. It also occasionally destroyed a "good" path, which necessitated our saving the best individuals from each generation, as mentioned previously. Mutation was implemented as follows: the mutatio probability μ determined whether it would occur within a given individual. If it did occur, a point along the path was selected at random. The remainder of the unmutated path between that point and the goal was destroyed. A new link to an allowed square was made, and the remainder of the path constructed using the random path generator. Clearly, most mutations had catastrophic effects for the fitness value of the mutant, however it did permit new material to evolve, which was not present in the original generation.

A final technique was instigated to accelerate the rate at which "good" paths evolved. The best member, (or members), of each generation is preserved intact for subsequent generations. Deified individuals are free from the threat of mutation, but do remain within the mating pool. The result of this non-biological device is a more rapid convergence toward the "good" solutions, and in the presence of high mutations rates, the necessary ingredient for <u>any</u> convergence.

Experimental Procedure:

The Genetic Algorithm program was developed on a Symbolics 3670, using Zeta LISP. Because the Genetic Algorithm is based upon manipulation of ordered lists, LISP is eminently suitable for this task. For the initial version of the program, a grid board was laid out, upon which the user could define obstacles, as well as elevations.

Following the selection of a specific board, or the construction thereof using the interactive graphics routines available upon the Symbolics, the initial conditions and parameters for the run were selected. These included population size, number of matings/per generation, number of culled individuals, number of "saved" individuals, and mutation rate; as well as the number of generations to be run. The run was then performed, recording the following information: the best path for each generation, the fitness value for that path, the average fitness for all paths of each generation, and the diversity of each generation. These data could be displayed graphically, as illustrated in Figure 3.

A series of runs were made, using a variety of boards and initial conditions. These could be grouped into three major classes: "Elitist", "Universalist", and "Radio-chemical Wastedump". The Elitist" strategy permitted only the most fit members of the generation to mate, as is characterized by high cull numbers and low mating number. It should be recalled that the individuals paired for mating are obtained by a weighted random selection process, and therefore no guarantee exists that the ones with the best fitness will be chosen. Not surprisingly, the number of generations (and hence computer time) necessary to obtain the good paths was greater for this strategy.

An improvement in the convergence was observed by using the "Universalist" strategy: high mating numbers, (usually involving all individuals within each generation), and low cull numbers. Even with the pairing of very good and very bad solutions, the average fitness improved more rapidly than in the case of the "Elitist" strategy.

The "Radio-chemical Wastedump" runs were characterized by high mutation rates. The high rate was imposed upon both the Elitist and Universalist strategies, with the result that convergence to a good solution disappeared. The average fitness values for these runs generally showed no improvement with increasing generation number. It was possible, however, to force improvement by imposing "deification"upon the best individual(s) from each generation; and when this is done, mutation behaves like genetic crossover. The average and best fitness values do improve with time, when both conditions are applied.

It is clear from the latter, that by imposing deification even without the high mutation rate, the convergence would be improved as well; and this was done to decrease the amount of computer time.

Results:

In order to determine the characteristics of the Path Planner, a series of runs were made for increasing complex terrain maps. For each terrain map, one or more of the following input parameters was varied: population size, number of mating pairs, cull number, mutation probability, and number of deified individuals per generation. The output from each set was examined to determine relative convergence rates for good solutions, and how "good" the solutions were compared to the "best" solution for that terrain map.

The initial set was run using a 10X10 square board, (Figure 4) with no variation in elevation. The start position was square (0, 0) and the goal was square (9, 9). The best path had a length (fitness value) of 13.899, and of the thirteen runs using this map, seven found this particular path, and the remaining six had fitness values less than 16.8. The ratio of mating pairs to population size was varied from 0.05 to 0.50. The variation in the convergence to good solutions between these runs was of the same magnitude as that observed in repeated runs using the same input set. Therefore, although there was a slight

indication that a ratio between 0.3 and 0.4 produced better convergence, it could not be isolated from the variation produced just by the random number generator.

In all runs in this set, the mutation probability was held at 0.05, and there were no individuals deified. Rapid convergence to good solutions were seen throughout; in a majority of cases, the best solutions was obtained. No definite trend was observed between Elitist and Universal mating strategies.

Two additional runs were made at this time, examining separate issues. These runs utilized a simple map, consisting of a 10X10 square board with a barracade between the start and goal. The first of these barracade runs was to examine the rate of convergence given a much larger set of possible initial paths. The concern was that good solutions would be much harder to obtain because the initial paths could be much more complex with the additional open spaces. This was found not to be the case. Convergence to good solutions occurred rapidly.

The second barracade run involved raising the mutation probability to 0.65, without saving any individuals. The average fitness and best fitness value for each generation failed to improve. It was found subsequently, that with a high mutation probability it was absolutely necessary to deify one or more individuals in each generations, in order for the best fitness value to improve; and even then the average fitness values failed to converge.

Following these runs, a more complex map was introduced, (Figure 5). This map consisted of 15X15 squares, with scattered obstacles, but still no variation in elevation. A total of eight runs were performed using this map. Again, the major issue was to determine whether there was any significant variation in rate of convergence to good solutions as the ratio between number of mating pairs and population size was varied. Two runs also involved adjusting the mutation probability.

The absolute best fitness (shortest path) for this map was calculated to be 20.97, and results for the eight runs varied between 21.5 and 32.5. Clearly, the results were not as satisfactory with this set as they were with the simpler map. However, convergence was observed in all cases, including that which consisted of mutation probability of 0.90. In fact, this case, in which a single individual was saved in each generation, recorded the best solution for the entire set, although the average fitness did not improve during the entire run. Again, no significant differences were observed between Elitist and Universal mating strategies.

A final run was made with this map utilizing a very low mutation probability (μ = 0.0001). It was observed that the best solution converged fairly rapidly to 23.3, or about 15% above the absolute best value. However, the diversity collapsed to near zero midway through the run, and consequently no further improvement could occur after that point. This illustrates the necessity of a small but finite mutation probability for the successful operation of the algorithm.

The next stage consisted of adding topology into the calculation of the fitness value, and illustrating it on the Symbolics Computer. The small number in the upper left hand corner of each square in Figure 6a represents the "elevation" of that square. Contour lines have been drawn in Figure 6b for better visualization. Represented are: a steep rising slope, a hill, a hole (or crater), and a ditch or "wash". the penalty imposed by climbing up a slope is equal to the square of the local slope. There is no penalty for climbing down.

Thirteen runs were performed using this terrain map. Calculating the absolute best path was extremely difficult; in fact, what is believed to be the best path was found by correcting one of the paths located by the genetic algorithm itself. The fitness value of this corrected "best" path is 23.314, and the best fitness values obtained during the runs ranged between 25.3 and 43.8, (average = 35.11, 51% above best value). The major area of investigation in this set focused upon the effect of deification upon the convergence to the "best"solution. By saving at least one member of each generation, it is guaranteed that the best fitness value solution is at least no worse during subsequent generations. A significant improvement was observed by the deification of multiple individuals within a generation, up to that point at which the genetic diversity collapsed. The best results were obtained with 5-10 individuals saved out of a generation size of 250. It was also observed that slightly better results were obtained using a mating pair number of 0.3 X generation size, although it is not clear that this is statistically significant.

Conclusions:

We have developed a path planner using a genetic algorithm. The principal differences between our algorithm and other genetic algorithms are: 1) the variable length of the list, 2) the way in which we performed crossover and mutation operations, and 3) the use of deification. It should be pointed out that a "greedy Algorithm" was available for post-processing, to straighten out kinks in the final paths, if so desired. None of the results discussed in this paper included that technique. In a real world case, such as onboard an autonomous planetary surface vehicle, this type of post-processing would doubtlessly be employed.

We make the following conclusion based upon our data:

1) The genetic Algorithm can be used to construct a robust path planner. Convergence to good solutions were obtained for a wide range of input parameters.

2) The inclusion of Deification improves the performance of the algorithm significantly.

3) Deification is required for convergence when using high mutation probabilities.

4) There is an indication that the optimum ratio of mating pairs to population size lies between the Elitist and Universal mating strategies. This value appears to be in the range 0.3 - 0.4, which implies up to 80% of the population involved in the genetic crossover operation.

It should finally be observed that one of the major drawbacks of the A* algorithm, which is used in many of the existing path planning programs, is that the search-tree is exponential. The corresponding search tree used in this Genetic Algorithm is of order N X P, where N is the number of nodes in the list, and P is the population size. This implies that the Genetic Algorithm is a far less computation intensive approach to path planning.

Reference:

[1] Goldberg, David E. "The Genetic Algorithm Approach: Why, How, and What Next", ADAPTIVE AND LEARNING SYSTEMS, Plenum Publishing Corp., 1986, pp 247-253

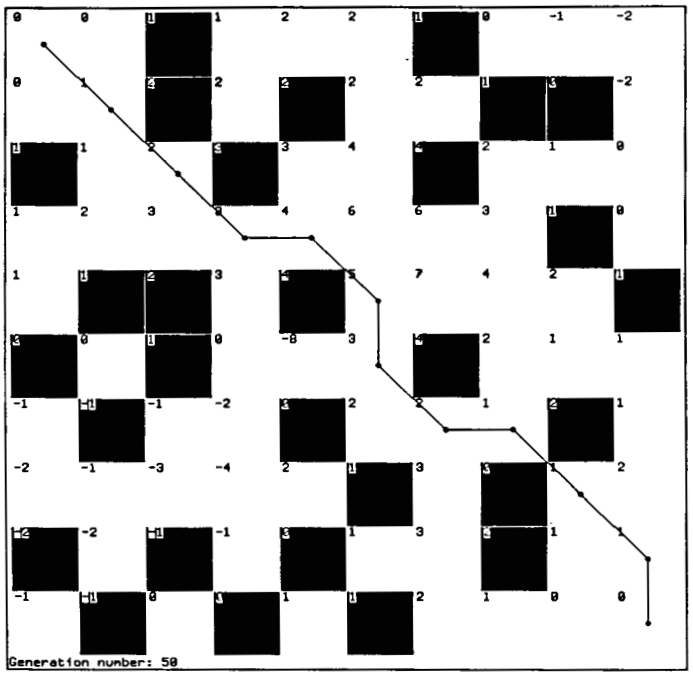

Figure 1. A 10 x 10 terrain map, illustrating obstacles, elevations, and a typical path

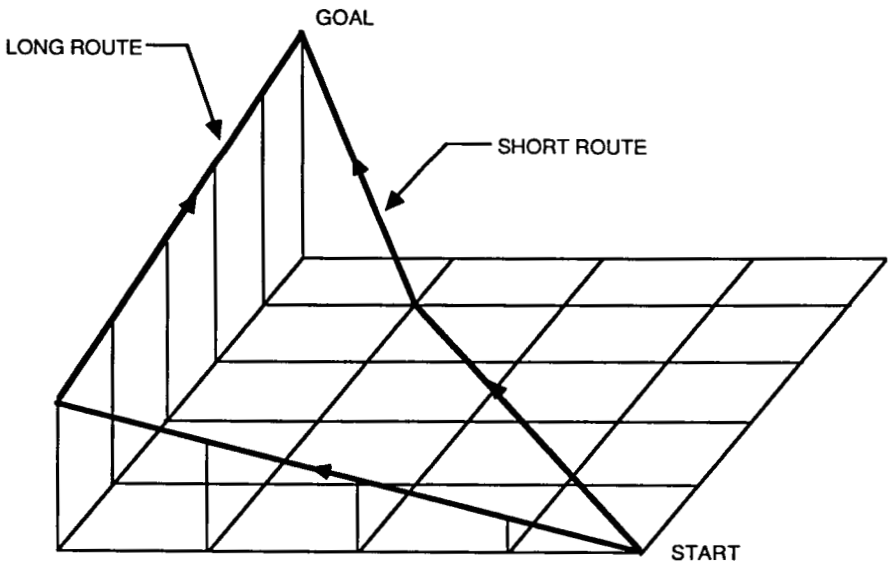

Figure 2. Paths illustrating gradual and steep slopes

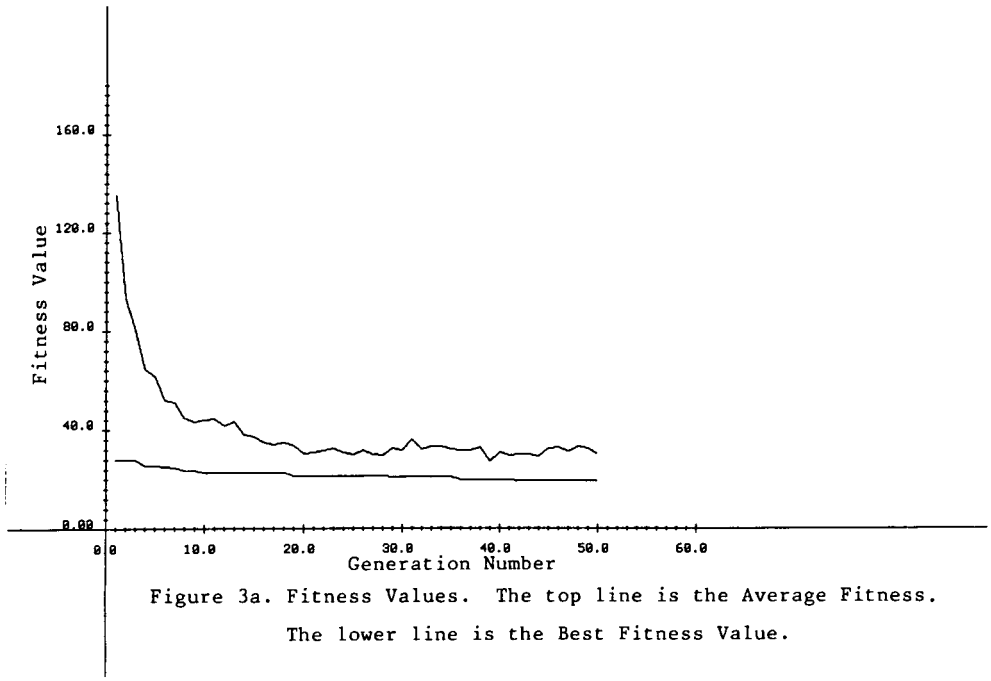

Figure 3a. Fitness Values. The top line is the Average Fitness. The lower line is the Best Fitness Value.

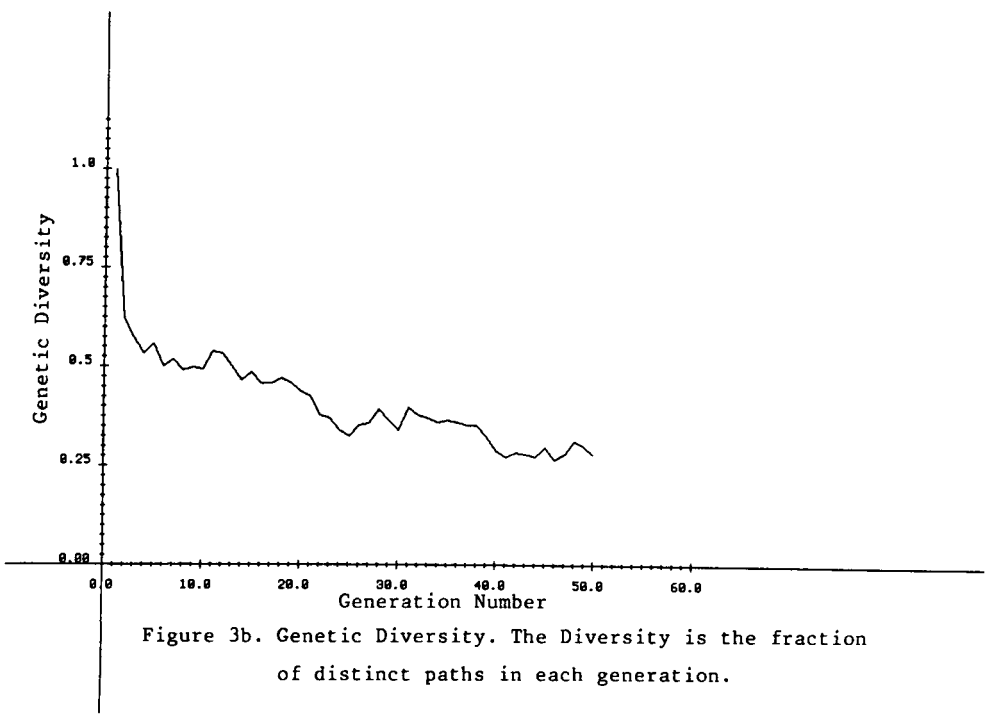

Figure 3b. Genetic Diversity. The Diversity is the fraction of distinct paths in each generation.

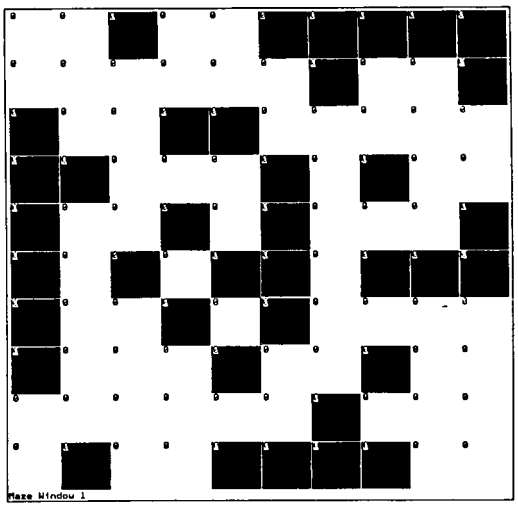

Figure 4. A flat 10 x 10 obstacle map

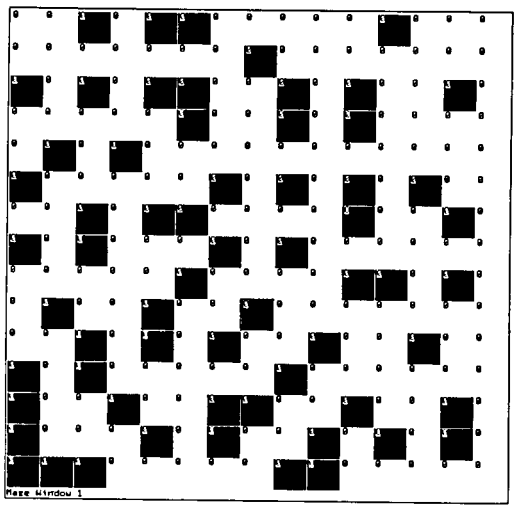

Figure 5. A flat 15 x 15 complex obstacle map

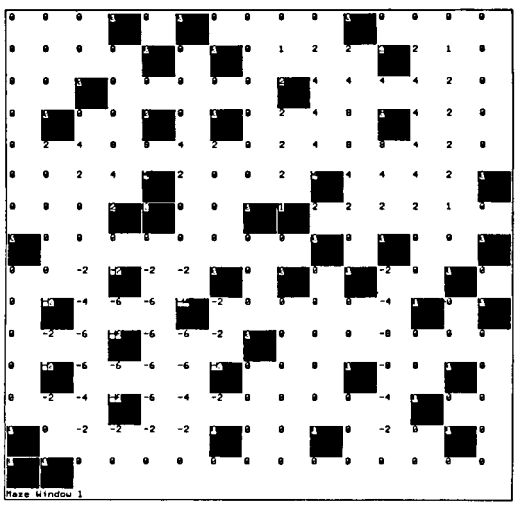

Figure 6a. The 15 x 15 complex terrain map, showing both positive and negative topology.

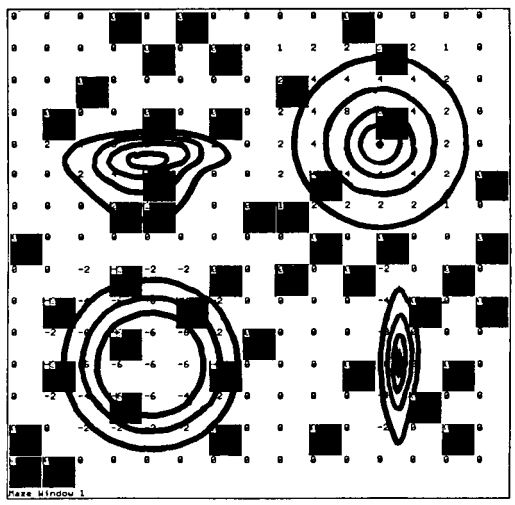

Figure 6b. The same complex terrain map with contour lines at 2 unit intervals.